Create Interactive Data Visualization with Plotly in R

2/06/2020

# Overview

* This is an RMarkdown explaining how we can use **“Swimmer Plots”** to visualize survival data.
* A **“Swimmer Plot”** is a graphical way of displaying several aspects of a subject’s tumor response such as **total time to tumor response**, whether there was a “Complete” or “Partial” response, and duration of response.
* This is a clear, graphical representation of the course of a patient’s tumor response and can be an especially useful tool when reporting clinical trial data results.

# Prepare the Data

## Step 1, Download ggplot2, reshape2, dplyr, plotly, and grid from CRAN

* Use the install.package() function to install the followng R packages from CRAN: ggplot2, plotly, reshape2, dplyr, kintr and grid from [CRAN](https://cran.r-project.org/) for example:  
  install.packages("ggplot2")

## Step 2, Load each relevant Package

library(ggplot2)   
library(dplyr, warn.conflicts=FALSE) # Useful for manipulating the dataframes  
library(reshape2) # Reformmating dataframes  
library(grid)  
library(plotly) # Allows us to make the swimmer plot interactive  
library(knitr)

## Step 3, Create an “example” data set for demonstrative purposes

* We will create a working data set appropriate for this type of graphical represenataion.

set.seed(35) # This sets the seed of R's random number generator  
dat <- data.frame(Subject = 1:15,   
 Months = sample(5:20, 15, replace=TRUE), # This generates a random set of months from 5 - 20  
 Treated=sample(0:1, 15, replace=TRUE), # This generates 15 random 0 or 1s which correspond to Tx or no Tx  
 Stage = sample(1:4, 15, replace=TRUE), # This randomly generates staging from 1 - 4   
 Continued=sample(0:15, 15, replace=TRUE))

### View initial Data Set

dat %>% kable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subject | Months | Treated | Stage | Continued |
| 1 | 14 | 0 | 4 | 5 |
| 2 | 10 | 0 | 2 | 9 |
| 3 | 12 | 0 | 1 | 0 |
| 4 | 5 | 0 | 1 | 14 |
| 5 | 11 | 0 | 2 | 10 |
| 6 | 15 | 1 | 4 | 1 |
| 7 | 13 | 0 | 1 | 13 |
| 8 | 18 | 1 | 1 | 9 |
| 9 | 9 | 0 | 2 | 0 |
| 10 | 6 | 0 | 4 | 6 |
| 11 | 6 | 0 | 3 | 2 |
| 12 | 20 | 0 | 1 | 15 |
| 13 | 13 | 0 | 2 | 2 |
| 14 | 14 | 1 | 1 | 12 |
| 15 | 13 | 0 | 1 | 0 |

## Add Response Data to Data Set

dat <- dat %>%  
 group\_by(Subject) %>%  
 mutate(Complete=sample(c(4:(max(Months)-1),NA), 1,   
 prob=c(rep(1, length(4:(max(Months)-1))),5), replace=TRUE),  
 Partial=sample(c(4:(max(Months)-1),NA), 1,   
 prob=c(rep(1, length(4:(max(Months)-1))),5), replace=TRUE),  
 Durable=sample(c(-0.5,NA), 1, replace=TRUE))  
# of note, `sample()`takes a sample of the specified size from the elements of x using either with or without replacement  
# Let's organize the order of the Subjects by Months  
dat$Subject <- factor(dat$Subject, levels=dat$Subject[order(dat$Months)])

### Let’s view the Data Set Now

dat %>% kable

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Subject | Months | Treated | Stage | Continued | Complete | Partial | Durable |
| 1 | 14 | 0 | 4 | 5 | 5 | 11 | NA |
| 2 | 10 | 0 | 2 | 9 | NA | 4 | -0.5 |
| 3 | 12 | 0 | 1 | 0 | 5 | 8 | -0.5 |
| 4 | 5 | 0 | 1 | 14 | NA | NA | NA |
| 5 | 11 | 0 | 2 | 10 | 5 | NA | -0.5 |
| 6 | 15 | 1 | 4 | 1 | 6 | 6 | -0.5 |
| 7 | 13 | 0 | 1 | 13 | 11 | 10 | -0.5 |
| 8 | 18 | 1 | 1 | 9 | 15 | 10 | NA |
| 9 | 9 | 0 | 2 | 0 | NA | 6 | -0.5 |
| 10 | 6 | 0 | 4 | 6 | NA | NA | -0.5 |
| 11 | 6 | 0 | 3 | 2 | 4 | NA | -0.5 |
| 12 | 20 | 0 | 1 | 15 | 11 | NA | NA |
| 13 | 13 | 0 | 2 | 2 | 8 | 9 | -0.5 |
| 14 | 14 | 1 | 1 | 12 | 13 | 13 | NA |
| 15 | 13 | 0 | 1 | 0 | NA | NA | -0.5 |

## Melt part of data frame for adding points to bars

* This will collapse the Columns “Complete”, “Partial” and “Durable” into a new column called “variable” and the values of those orginial columns will become a new vector/column called “value”

dat.m <- melt(dat %>% select(Subject, Months, Complete, Partial, Durable),  
 id.var=c("Subject","Months"), na.rm = TRUE)   
# of note, na.rm = TRUE will eliminate those rows with missing values

### Let’s View our Data Set after melting

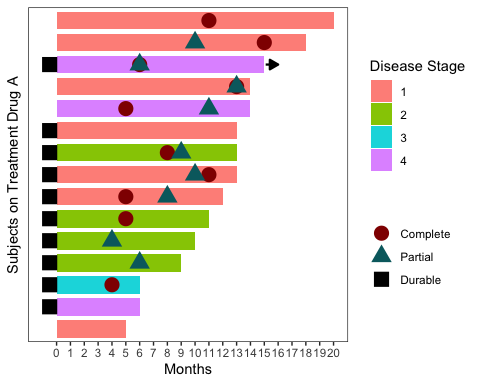
dat.m %>% kable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Subject | Months | variable | value |
| 1 | 1 | 14 | Complete | 5.0 |
| 3 | 3 | 12 | Complete | 5.0 |
| 5 | 5 | 11 | Complete | 5.0 |
| 6 | 6 | 15 | Complete | 6.0 |
| 7 | 7 | 13 | Complete | 11.0 |
| 8 | 8 | 18 | Complete | 15.0 |
| 11 | 11 | 6 | Complete | 4.0 |
| 12 | 12 | 20 | Complete | 11.0 |
| 13 | 13 | 13 | Complete | 8.0 |
| 14 | 14 | 14 | Complete | 13.0 |
| 16 | 1 | 14 | Partial | 11.0 |
| 17 | 2 | 10 | Partial | 4.0 |
| 18 | 3 | 12 | Partial | 8.0 |
| 21 | 6 | 15 | Partial | 6.0 |
| 22 | 7 | 13 | Partial | 10.0 |
| 23 | 8 | 18 | Partial | 10.0 |
| 24 | 9 | 9 | Partial | 6.0 |
| 28 | 13 | 13 | Partial | 9.0 |
| 29 | 14 | 14 | Partial | 13.0 |
| 32 | 2 | 10 | Durable | -0.5 |
| 33 | 3 | 12 | Durable | -0.5 |
| 35 | 5 | 11 | Durable | -0.5 |
| 36 | 6 | 15 | Durable | -0.5 |
| 37 | 7 | 13 | Durable | -0.5 |
| 39 | 9 | 9 | Durable | -0.5 |
| 40 | 10 | 6 | Durable | -0.5 |
| 41 | 11 | 6 | Durable | -0.5 |
| 43 | 13 | 13 | Durable | -0.5 |
| 45 | 15 | 13 | Durable | -0.5 |

# Graph the Data using a Swimmer Plot

## Let’s make a static swimmer plot with ggplot

a<- ggplot(dat, aes(Subject, Months)) +  
 geom\_bar(stat="identity", aes(fill=factor(Stage)), width=0.8) +  
 geom\_point(data=dat.m,   
 aes(Subject, value, colour=variable, shape=variable), size=5) +  
 geom\_segment(data=dat %>% filter(Continued==1),   
 aes(x=Subject, xend=Subject, y=Months + 0.1, yend=Months + 1),   
 pch=15, size=0.8, arrow=arrow(type="closed", length=unit(0.1,"in"))) +  
 coord\_flip() +  
 scale\_fill\_manual(values=hcl(seq(15,375,length.out=5)[1:4],100,75)) +  
 scale\_colour\_manual(values=c(hcl(seq(15,375,length.out=3)[1:2],100,30),"black")) +  
 scale\_y\_continuous(limits=c(-1,20), breaks=0:20) +  
 labs(fill="Disease Stage", colour="", shape="",  
 x="Subjects on Treatment Drug A") +  
 theme\_bw() +  
 theme(panel.grid.minor=element\_blank(),  
 panel.grid.major=element\_blank(),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank())  
a



## Now let’s make an Interactive Swimmer plot in Plotly by simply using the ggplotly() function of the static plot as an object

ggplotly(a)

# SessionInfo

sessionInfo()

## R version 3.6.1 (2019-07-05)  
## Platform: x86\_64-apple-darwin15.6.0 (64-bit)  
## Running under: macOS Mojave 10.14.6  
##   
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib  
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib  
##   
## locale:  
## [1] en\_US.UTF-8/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8  
##   
## attached base packages:  
## [1] grid stats graphics grDevices utils datasets methods   
## [8] base   
##   
## other attached packages:  
## [1] knitr\_1.26 plotly\_4.9.0 reshape2\_1.4.3 dplyr\_0.8.3 ggplot2\_3.2.1   
##   
## loaded via a namespace (and not attached):  
## [1] Rcpp\_1.0.3 later\_1.0.0 highr\_0.8 pillar\_1.4.3   
## [5] compiler\_3.6.1 plyr\_1.8.5 tools\_3.6.1 digest\_0.6.23   
## [9] viridisLite\_0.3.0 jsonlite\_1.6 evaluate\_0.14 lifecycle\_0.1.0   
## [13] tibble\_2.1.3 gtable\_0.3.0 pkgconfig\_2.0.3 rlang\_0.4.4   
## [17] shiny\_1.4.0 crosstalk\_1.0.0 yaml\_2.2.0 xfun\_0.11   
## [21] fastmap\_1.0.1 withr\_2.1.2 stringr\_1.4.0 httr\_1.4.1   
## [25] vctrs\_0.2.2 htmlwidgets\_1.5.1 tidyselect\_0.2.5 glue\_1.3.1   
## [29] data.table\_1.12.6 R6\_2.4.1 rmarkdown\_1.17 farver\_2.0.3   
## [33] tidyr\_1.0.0 purrr\_0.3.3 magrittr\_1.5 promises\_1.1.0   
## [37] scales\_1.1.0 htmltools\_0.4.0 assertthat\_0.2.1 xtable\_1.8-4   
## [41] mime\_0.7 colorspace\_1.4-1 httpuv\_1.5.2 stringi\_1.4.5   
## [45] lazyeval\_0.2.2 munsell\_0.5.0 crayon\_1.3.4